Module 4 – Pivoting and Advanced Grouping

Overview

The main purpose of this lab is to familiarise yourself with how to use pivot and the advanced grouping clauses within TSQL.

- Use PIVOT to translate rows into columns
- Aggregations using ROLLUP and GROUPING SETS

Objectives

At the end of this lab, you will be able to:

- Write a query to utilise the PIVOT clause
- Use advanced grouping techniques like ROLLUP and GROUPING SETS

Setup: Launch SQL Server Management Studio

1. On the Start menu, click All Programs, click Microsoft SQL Server 2014 and then click SQL Server Management Studio.
2. The Microsoft SQL Server Management Studio window opens, and then the Connect to Server dialog box will appear.
3. In the Connect to Server dialog box, click Connect to accept the default settings.
4. On the toolbar, click New Query, and either select the QATSQLPLUS database in the Available Databases box, or type USE QATSQLPLUS in the query window.
5. If you wish, you may save your queries to My Documents or the desktop. All modules are separate and you will not require any queries from this module in any later module.

Exercise 1: Attendance by Date and Vendor

In this exercise, you will create a query to return a matrix-like result using the VendorCourseDateDelegateCount view.

The main tasks for this exercise are as follows:

1. Create a query to review the content of the dbo.VendorCourseDateDelegateCount view.
2. Create a query to return the pivoted result with VendorName values as columns.

Task 1: Review table content

1. Write a query to return the dbo.VendorCourseDateDelegateCount view. The columns returned should be:
   - VendorName
   - CourseName
   - StartDate
   - NumberDelegates
2. Test the query. The query should return 9 rows.
3. Keep the query window open for the following tasks.

Task 2: Pivot the resultset

1. Write a query to return the dbo.VendorCourseDateDelegateCount view. The columns returned should be:
   - VendorName
   - StartDate
   - NumberDelegates
2. Test the query. The query should return 9 rows.
3. Change the query above into a derived table in the form:

   ```
   SELECT * FROM
   (<the query from step 1>) AS BaseData
   ```

4. Test the query. The query should return 9 rows. This should be exactly the same result as the previous query result.
5. Change the query from step 3 to PIVOT the Vendor names and aggregate the total of NumberDelegates.

6. Test the query. The query should return 3 rows with 4 columns as shown below:

	StartDate	QA	Microsoft	Oracle
1	2016-10-03	8	NULL	2
2	2016-10-10	NULL	2	NULL
3	2016-10-13	3	NULL	NULL
4	2016-10-17	5	NULL	NULL
5	2016-10-24	2	17	NULL
6	2016-10-26	3	NULL	NULL
7	2016-10-31	NULL	7	NULL

7. Keep the query window open for the following tasks.

Exercise 2: Custom Subtotals

In this exercise, you find produce a summary of number of delegates with custom subtotals.

The main tasks for this exercise are as follows:

1. Create a query that returns list of events with a subtotal for VendorName, CourseName and StartDate.
2. Create a query that returns list of events with a subtotal for only VendorName, and CourseName.

Task 1: Using ROLLUP

1. Write a query to return all columns from the dbo.VendorCourseDateDelegateCount view. The columns returned should be:
   - VendorName
   - CourseName
   - StartDate
   - NumberDelegates
2. Test the query. The query should return 9 rows.
3. Change the query so that the NumberDelegates is aggregated using the SUM function, keeping the other columns in the SELECT list. Add a GROUP BY clause as appropriate.
4. Test the query. The query should still return 9 rows.
5. Change the query to add the WITH ROLLUP clause. Unlike most clauses in TSQL, the order of columns listed in the GROUP BY does matter when using ROLLUP.

6. Test the query. The query should return 20 rows. If the number of rows is not 20 then recheck the order of the columns in the GROUP BY clause. Extra rows will be added to summarise the combinations of:
    - VendorName, CourseName, StartDate
    - VendorName, CourseName
    - VendorName
    - Total

	Vendorname	CourseName	StartDate	TotalDelegates
1	Microsoft	Administering a SQL Database Infrastructure (2016)	2016-10-24	17
2	Microsoft	Administering a SQL Database Infrastructure (2016)	NULL	17
3	Microsoft	Developing SQL Data Models (2016)	2016-10-31	7
4	Microsoft	Developing SQL Data Models (2016)	NULL	7
5	Microsoft	Querying data with Transact-SQL (2016)	2016-10-10	2
6	Microsoft	Querying data with Transact-SQL (2016)	NULL	2
7	Microsoft	NULL	NULL	26
8	Oracle	mySQL for Developers	2016-10-03	2
9	Oracle	mySQL for Developers	NULL	2
10	Oracle	NULL	NULL	2
11	QA	QASQLRB08	2016-10-26	3
12	QA	QASQLRB08	NULL	3
13	QA	QATSQL	2016-10-03	8
14	QA	QATSQL	2016-10-13	3
15	QA	QATSQL	NULL	11
16	QA	QATSQLPLUS	2016-10-17	5
17	QA	QATSQLPLUS	2016-10-24	2
18	QA	QATSQLPLUS	NULL	7
19	QA	NULL	NULL	21
20	NULL	NULL	NULL	49

7. In the picture above the rows 2,4,6,9,12,15,18 show the total for VendorName and CourseName regardless of the StartDate. Row 20 shows the grand total. If these rows were not required, could they be removed easily?
8. Keep the query window open for the following tasks.

Task 2: Use GROUPING SETS

1. Write a query to return all columns from the dbo.VendorCourseDelegateCount view. The columns returned should be:
    - VendorName
    - CourseName
    - StartDate
    - NumberDelegates
2. Test the query. This should return 9 rows.
3. Change the query so that the NumberDelegates is aggregated using the SUM function, keeping the other columns in the SELECT list. Add a GROUP BY clause as appropriate.
4. Test the query. This should still return 9 rows.
5. Change the query to add additional rows for the subtotals for Vendor only and VendorName, CourseName and StartDate. Hint: Grouping Sets.
6. Test the query. This should return 12 rows. An additional row should be added for each VendorName (with a vendor total).
7. Change the query to add subtotal rows for each combination of VendorName and CourseName. Add an additional column showing which shows which columns have been aggregated. HINT: Grouping_ID.
8. Test the query. This should return 19 rows. The grouping column should show 0, 1 or 3. The expected result is shown below.

	Vendorname	CourseName	StartDate	TotalDelegates	GroupingType
1	Microsoft	Administering a SQL Database Infrastructure (2016)	2016-10-24	17	0
2	Microsoft	Administering a SQL Database Infrastructure (2016)	NULL	17	1
3	Microsoft	Developing SQL Data Models (2016)	2016-10-31	7	0
4	Microsoft	Developing SQL Data Models (2016)	NULL	7	1
5	Microsoft	Querying data with Transact-SQL (2016)	2016-10-10	2	0
6	Microsoft	Querying data with Transact-SQL (2016)	NULL	2	1
7	Microsoft	NULL	NULL	26	3
8	Oracle	mySQL for Developers	2016-10-03	2	0
9	Oracle	mySQL for Developers	NULL	2	1
10	Oracle	NULL	NULL	2	3
11	QA	QASQLRB08	2016-10-26	3	0
12	QA	QASQLRB08	NULL	3	1
13	QA	QATSQL	2016-10-03	8	0
14	QA	QATSQL	2016-10-13	3	0
15	QA	QATSQL	NULL	11	1
16	QA	QATSQLPLUS	2016-10-17	5	0
17	QA	QATSQLPLUS	2016-10-24	2	0
18	QA	QATSQLPLUS	NULL	7	1
19	QA	NULL	NULL	21	3

9. The query window can be closed. Save the query if you wish.

Answers

The answers below are for example only. Coding style and order of columns should not matter.

Ex 1 Task 1

```
--Task 1:
SELECT *
  FROM dbo.VendorCourseDateDelegateCount
```

Ex 1 Task 2

```
--Task 2a:
SELECT VendorName, StartDate, NumberDelegates
  FROM dbo.VendorCourseDateDelegateCount

--Task 2b:
SELECT *
  FROM (SELECT VendorName, StartDate, NumberDelegates
    FROM dbo.VendorCourseDateDelegateCount) AS BaseData

--Task 2c:
SELECT *
  FROM (SELECT VendorName, StartDate, NumberDelegates
    FROM dbo.VendorCourseDateDelegateCount) AS BaseData
  PIVOT
  (SUM(NumberDelegates) FOR VendorName IN (QA, Microsoft, Oracle))
    AS Pivotted
```

Ex 2 Task 1

```
--Task 1a:
SELECT Vendorname, CourseName, StartDate, NumberDelegates
  FROM dbo.VendorCourseDateDelegateCount

--Task 1b:
SELECT Vendorname, CourseName, StartDate,
    SUM(NumberDelegates) AS TotalDelegates
  FROM dbo.VendorCourseDateDelegateCount
  GROUP BY Vendorname, CourseName, StartDate
GO

--Task 1c:
SELECT Vendorname, CourseName, StartDate,
    SUM(NumberDelegates) AS TotalDelegates
  FROM dbo.VendorCourseDateDelegateCount
  GROUP BY Vendorname, CourseName, StartDate
    WITH ROLLUP
GO
```

Ex 2 Task 2

```
--Task 2a:
SELECT Vendorname, CourseName, StartDate, NumberDelegates
  FROM dbo.VendorCourseDateDelegateCount


--Task 2b:
SELECT Vendorname, CourseName, StartDate,
    SUM(NumberDelegates) AS TotalDelegates
  FROM dbo.VendorCourseDateDelegateCount
  GROUP BY GROUPING SETS (
    (VendorName),
    (Vendorname, CourseName, StartDate)
  )
GO
--Task 2c:
SELECT Vendorname, CourseName, StartDate,
    SUM(NumberDelegates) AS TotalDelegates,
    GROUPING_ID(VendorName, Coursename, StartDate) AS GroupingType
  FROM dbo.VendorCourseDateDelegateCount
  GROUP BY GROUPING SETS (
    (VendorName),
    (Vendorname, CourseName),
    (Vendorname, CourseName, StartDate)
  )
GO
```